A MFoM Learning Approach to Robust Multiclass Multi-Label Text Categorization

Sheng Gao GAOSHENG@I2R.A-STAR.EDU.SG
Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613

Wen Wu WENWU@CS.CMU.EDU
Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. 15213, USA

Chin-Hui Lee CHL@ECE.GATECH.EDU
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA. 30332, USA

Tat-Seng Chua CHUATS@COMP.NUS.EDU.SG
School of Computing, National University of Singapore, 3 Science Drive 2, Singapore, 117543

Abstract

We propose a multiclass (MC) classification approach to text categorization (TC). To fully take advantage of both positive and negative training examples, a maximal figure-of-merit (MFoM) learning algorithm is introduced to train high performance MC classifiers. In contrast to conventional binary classification, the proposed MC scheme assigns a uniform score function to each category for each given test sample, and thus the classical Bayes decision rules can now be applied. Since all the MC MFoM classifiers are simultaneously trained, we expect them to be more robust and work better than the binary MFoM classifiers, which are trained separately and are known to give the best TC performance. Experimental results on the Reuters-21578 TC task indicate that the MC MFoM classifiers achieve a micro-averaging F_1 value of 0.377, which is significantly better than 0.138, obtained with the binary MFoM classifiers, for the categories with less than 4 training samples. Furthermore, for all 90 categories, most with large training sizes, the MC MFoM classifiers give a micro-averaging F_1 value of 0.888, better than 0.884, obtained with the binary MFoM classifiers.

1. Introduction

Text categorization (TC) is a process of classifying a text document into some pre-defined categories. It is an important research problem in information retrieval (IR), information extraction and filtering, web text mining, and natural language processing. In the past two decades TC has received much attention (Sebastiani, 2002). It is usually formulated as the binary classification (BC) that a binary classifier is designed for each category of interest, and multiclass, multi-label decisions are made based on how well the input text document matches with the set of multiple binary classifiers (Sebastiani, 2002). Although the BC approach has achieved high performance in TC, only two-class classification is considered. On the other hand, we have also seen the multiclass (MC) classification approach being used in many pattern recognition problems, such as automatic speech recognition, speaker identification, face recognition and optical character recognition. Here by MC classification, we mean a uniform score function is defined to compute a goodness-of-fit score between the input document and a category-specific classifier. Multi-label classification can be accomplished by selecting a group of top scoring categories. If a source distribution can be attached to each category, then the Bayes decision theory serves as the theoretical foundation for the multiclass classification. A lot of literatures on this topic are available (e.g. Lee & Huo, 2000).

Numerous training algorithms have been studied in BC, e.g. Naïve Bayesian method (Lewis & Ringuette, 1994; Yang & Liu, 1999), *k*-nearest neighbors (*k*NN) (Yang & Liu, 1999), support vector machines (SVM) (Joachims, 2002), decision tree (Breiman, *et al.*1984), etc. To handle multiclass classification, error-correcting output coding (ECOC) (Dietterich & Bakiri, 1995), boosting (Schapire & Singer, 2000), and multiclass SVM (Weston & Watkins, 1999; Vapik 1998; Zhang, *et al.*, 2003) have been proposed. But only a handful of techniques are available for direct multiclass classification. Although some BC algorithms can be extended to learning multiclass, multi-label classifiers (e.g. Naïve Bayesian, *k*NN, decision tree), they do not work as well as the

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004.

binary classifiers commonly adopted in IR and TC communities.

To improve the performance on TC, both positive and negative examples need to be considered during training. This principle is implicitly used to learn the binary classifiers. However, discrimination among the categories is not fully utilized because these category-specific binary classifiers are usually trained independently. To mitigate this deficiency, a maximal figure-of-merit (MFoM) approach was introduced to learn the binary classifiers (Gao, *et al*, 2003). It was demonstrated that high performance TC systems can be designed using the MFoM-trained classifiers.

In this paper, we further generalize the MFoM learning to multiclass, multi-label classification. To learn these MC classifiers, any objective function of performance metrics, such as precision and recall, can be formulated and optimized. The proposed MFoM learning algorithm takes full advantage of both positive and negative examples. Since all MC MFoM classifiers are trained simultaneously, we expect them to be more robust than the corresponding binary MFoM classifiers, especially for the categories with a very small number of positive training examples. Our experimental results on the Reuters-21578 TC task indeed indicate that the MC MFoM classifiers achieve a micro-averaging F_1 value of 0.377, which is significantly better than 0.138, obtained with the binary MFoM classifiers, for those categories with less than 4 training samples. Furthermore, for all 90 categories, mostly with large training sizes, the MC MFoM classifiers give a macro-averaging F_1 value of 0.888, which is better than 0.884, obtained with the binary MFoM classifiers.

The rest of the paper is organized as follows. In Section 2, some related studies are first briefly described. The theory of MC MFoM learning is then presented in Section 3. Next in Section 4, we cast the MFoM learning into a baseline classifier with a linear discriminant function used in designing the binary MFoM classifiers (Gao, *et al.* 2003). In Section 5 we report on the experimental results with the Reuters-21578 TC task. Finally we conclude our findings in Section 6.

2. Related Studies

The goal of our MC MFoM learning approach is to design a discriminative multi-label multiclass classifier so that we can sufficiently maximize the pair-wise discrimination power among all competing classes (Remember most of them are ignored in learning binary classifier) and can explicitly optimize our preferred figure-of-merit. In the following we review some related studies.

A popular TC learning approach is based on a divide-and-conquer strategy, i.e. decomposing the multi-class classification problem into multiple binary classifiers and then combing the binary decisions if necessary. Two such examples are ECOC (Dietterich & Bakiri, 1995) and BoosTexter (Schapire & Singer, 2000). To train a binary classifier, the first step is to choose a suitable set of samples to estimate a particular binary classifier. In ECOC, the instance selection depends on a coding matrix, generally designed based on some expert knowledge (e.g. the entry value is 1 if an instance is positive. Otherwise, it is 0). While in the BoosTexter, the selection is based on the iteratively estimated joint distribution of the training samples and the category. The decision with multiple binary classifiers can be a majority voting (Dietterich & Bakiri, 1995), or a linear combination of the scores from all the weaker hypotheses (Schapire & Singer, 2000).

The above divide-and-conquer methodology has been quite successful because it explicitly captures the ability to discriminate between positive and negative examples. However each binary classifier is learned independently and we have no knowledge about how to compare two such binary classifiers. If all the pair-wise comparisons are taken into account in training, we should be able to further improve the discriminative power and robustness of the classifiers. Recent work on multiclass SVM (Weston & Watkins, 1999) showed that the learned classifier has less number of support vectors than the binary SVM, although there was no improvement in the performance. This implies a more compact representation of the decision surfaces for MC classifiers than the binary ones.

Classifier (binary or multiclass) learning can be done by optimizing an objective function which measures some predefined merits, such as an empirical classification error (Katagiri, *et al.*, 1998; Schapire & Singer, 2000), a regularized empirical classification error (Weston & Watkins, 1999; Joachims, 2002; Zhang, *et al.*, 2003), an empirical ranking loss (Schapire & Singer, 2000), etc. Most of the measures are not the desired metric for evaluating the system. In TC, the classification error cannot effectively measure the classifier, and the false positive and false negative rates or their combination (e.g. F_1) are more popular. To make the objective function effectively measure the classifiers, Yan, *et al.* (2003) applied the Wilcoxon-Mann-Whitney statistic, a measure of the overall pair-wise ranking error in the training set, as an objective function. Their experimental results showed that the area under the ROC curve is maximized, however, the trained classifiers are not guaranteed to be optimal at a particular operating point. In contrast to this study, Gao, *et al*. (2003) proposed a MFoM learning approach to optimize any preferred merit for any given classifiers. By a smooth embedding of the decision function, the overall nonlinear objective function is continuously differentiable, and can therefore be iteratively optimized with a generalized probabilistic descend algorithm (Katagiri, *et al*, 1998).

3. Multiclass MFoM Learning

In (Gao, *et al*., 2003), a binary MFoM learning approach is proposed to train the binary classifiers by optimizing any overall objective function of any metric of interest. It simulates the discrete performance measures (e.g. recall, precision, F_1) with a continuous function of the classifier parameters. Since the objective function is continuous and differentiable, it can be optimized to obtain the MFoM-trained binary classifiers. In the following we extend it to learn multiclass classifiers.

3.1 Single-Label Decision Rule

Given *N* categories, $C=\{C_j, 1\le j\le N\}$, and a multi-labeled training set, $T=\{(X,Y)|X\in R^D, Y\subset C\}$, where C_j is the *j*-th category, with *X* being a sample in a *D*-dimension space, *Y* representing a set of labels for *X* and a subset of *C*. *N* classifiers with the corresponding parameter set, Λ, are estimated from *T*. In the most general case, all the classifiers could share part or all of the parameters. In the current implementation, we assume that each classifier is defined as its own parameter set, Λ_j, for the *j*-th category, C_j, then $\Lambda=\{\Lambda_j, 1\le j\le N\}$. For most pattern recognition problems, such as speaker identification, fingerprint recognition, and optical character recognition, there is only one category label in *Y* associated with *X*. It is simply a decision to classify a given instance *X* into one of *N* categories. A popular choice of the decision rule is to decide on the category with the maximum score as the recognized class, i.e.

$$\hat{C}(X)=\arg\max_j g_j(X;\Lambda_j), 1\le j\le N \quad , \qquad (1)$$

where $\hat{C}(X)$ is the assigned category, and $g_j(X;\Lambda_j)$ a *class discriminant function* (or *score function*) to compute the class scores. In such multiclass, single-label classification cases, it can be shown that a correct classification decision is made for a sample, *X*, coming from the class C_j, if $\hat{C}(X)=C_j$, i.e. $\forall i$

$$g_j(X;\Lambda_j) > g_{i\ne j}(X;\Lambda_i) \quad X\in C_j \qquad (2)$$

which is equivalent to the decision rule in Eq. (1).

3.2 Multi-Label Decision Rule

Next we address the multi-label decision issue with the multiclass classification. First, a list of *N*-best categories is decided with the decision rule in Eq. (1). Then based on a confidence measure, we perform a test to verify if each candidate should be accepted or rejected. In the multi-label case, a test sample may come from a subset of the *N* categories. Correct classification is achieved only when all true labels are detected. If the true category is not in the list of the top ranked categories, we have a wrong decision. This makes evaluation a tricky problem.

In speech recognition (Katagiri, *et al.*, 1998) and call routing (Kuo & Lee, 2003), the scores from the most competing categories against a specific category is combined to measure the distance between the target and its decision surface. Here the same idea is applied to define a competitive model, called *class anti-discriminant function*,

$$g_j^-(X;\Lambda^-)=\log\left[\frac{1}{|C_j^-|}\sum_{i\in C_j^-}\exp\left(g_i(X;\Lambda_i)\right)^h\right]^{1/h} \qquad (3)$$

where C_j^-, often called a cohort set, is a subset containing the most competitive categories against C_j, $|C_j^-|$ is the cardinality of the cohort set, Λ^- is the parameter set for all the competitive categories, and h is a positive constant. The term in the RHS of Eq. (3) represents a geometric average of all the competing scores. Intuitively, it is noted that when h approaches ∞, the RHS of Eq. (3) converges to the highest score among the competing categories. With Eq. (3), the decision rule to verify each category can be formulated as:

$$\begin{cases}\text{Accept} & X\in C_j \text{ if } g_j(X;\Lambda_j)-g_j^-(X;\Lambda^-)>0\\ \text{Reject} & X\in C_j, \textit{Otherwise}\end{cases} \quad 1\le j\le N \qquad (4).$$

It is similar to the decision rule in the binary classification. Here the difference between $g_j(X;\Lambda_j)$ and $g_j^-(X;\Lambda^-)$ tries to measure the confidence of the decision in Eq. (4), and can be compared and ranked among all categories. This property provides a more efficient decision rule for the multiclass classification. For example, we only have to verify the top *N*-best category candidates according to their confidence rankings. Intuitively we expect the ranking of all correct categories to be higher than that of all other categories. In the training stage we attempt to maximize the separation between the correct and competing categories, by adjusting all the classifier parameters to move upward the rankings of the correct categories, and move downward the incorrect ones.

3.3 Overall Objective Function

Various objective functions have been proposed, e.g. the likelihood function in *ML* training (Katagiri, *et al.*, 1998), least square distance (Yang & Liu, 1999), empirical error count (Katagiri, *et al.*, 1998; Kuo & Lee, 2003), empirical ranking loss (Schapire & Singer, 2000), etc. In TC, it is important to explicitly define an overall objective function integrating all preferred performance metrics with any classifier.

The basic quantity for the performance measure is the classification error. Any other metrics, such as true positive (TP), false positive (FP), false negative (FN), precision, recall, and F_1, can be expressed as a function of error counts. We next define a continuous and differentiable function of the classifier parameters to simulate these error counts.

According to the decision rule in Eq. (4), we define a one-dimension class misclassification function, $d_j(X;\Lambda)$, such that Eq. (4) is equivalent to $d_j(X;\Lambda)<0$ when a correct decision is made, i.e. zero error increment. Otherwise, $d_j(X;\Lambda)\geq 0$. This can be accomplished with

$$d_j(X;\Lambda) = -g_j(X;\Lambda_j) + g_j(X;\Lambda_j^-) \quad . \tag{5}$$

The absolute value of $d_j(X;\Lambda)$ quantifies the separation between the correct and competing categories. So it can be treated as a confidence measure. If it is much less than zero, we are more confident on accepting the *j*-th category. On the other hand, if $d_j(X;\Lambda)$ is much larger than zero, we are more sure to reject the *j*-th category.

While the sign of the class misclassification value can indicate whether an instance is correctly detected or not, its relation to the error is still discrete. To simulate the error count with a continuous function, a class loss function, $l_j(X;\Lambda)$, for category C_j, is defined. It should be close to 0 for correct detections, and 1 for incorrect ones. Clearly this loss is a function of the feature vector, *X*, and the classifier parameters, Λ. As for the choice of an appropriate loss function, any smooth 0-1 function of a one-dimension variable approximating a step function at the origin will do the job. A sigmoid function is often adopted (e.g. Gao, *et al.*, 2003) as

$$l_j(X;\Lambda) = \frac{1}{1+e^{-a(d_j(X;\Lambda)+b)}} \quad , \tag{6}$$

where a is a positive constant that controls the size of the learning window and the learning rate, and b is a constant measuring the offset of $d_j(X;\Lambda)$ from 0. Its value can simulate the error count made by the *j*-th classifier for a given test sample *X*.

Denote the precision, recall, and F_1 measures for a class C_j by P_j, R_j, and F_j, respectively. We have

$$P_j = \frac{TP_j}{TP_j + FP_j} \quad , \tag{7}$$

$$R_j = \frac{TP_j}{TP_j + FN_j} \quad , \tag{8}$$

$$F_j = \frac{2P_jR_j}{R_j + P_j} = \frac{2TP_j}{FP_j + FN_j + 2TP_j} \quad . \tag{9}$$

It is clear that these metrics are discrete quantities for counting errors and could not be optimized directly because they are not differentiable functions of the parameters. To evaluate the classification performance for each category, precision, recall, and F_1 measure in Eqs.(7-9) are used. To evaluate the average performance over many categories, the macro-averaging F_1 and micro-averaging F_1 are used and defined as follows:

$$F_1^M = 2[\sum_{i=1}^N R_i \sum_{i=1}^N P_i] / N[\sum_{i=1}^N R_i + \sum_{i=1}^N P_i], \tag{10}$$

$$F_1^m = 2\sum_{i=1}^N TP_i / [\sum_{i=1}^N FP_i + \sum_{i=1}^N FN_i + 2\sum_{i=1}^N TP_i] \ , \tag{11}$$

i.e. the micro-averaging metric computes an overall global measure by giving different weights to each category's local performance measures based on their numbers of positive documents. Macro-averaging method treats every category equally, and calculates the global measures as the means of the local measures of all categories.

Now we can simulate these error counts in the training set *T* for each category C_j as follows:

$$FN_j \approx \sum_{X\in T} l_j(X;\Lambda)\cdot 1(X\in C_j) \ , \tag{12}$$

$$FP_j \approx \sum_{X\in T} (1-l_j(X;\Lambda))\cdot 1(X\notin C_j) \ , \tag{13}$$

$$TP_j \approx \sum_{X\in T} (1-l_j(X;\Lambda))\cdot 1(X\in C_j), \tag{14}$$

where *1(A)* is the indicator function of any logical expression, *A*. The above quantities are continuous and differentiable functions of the classifier parameters. Let $L(T;\Lambda)$ be an overall objective function, which depends on the above three quantities. We can solve for the classifier parameters by maximizing or minimizing the objective function in its general form, i.e.

$$L(T;\Lambda) = f(TP_j, FN_j, FP_j | 1\leq j\leq N) \tag{15}$$

3.4 GPD Algorithm to Learn Classifier Parameters

In most cases, the objective function defined in Eq. (15) is highly nonlinear. A generalized probabilistic descent (GPD) algorithm was proposed to find its solution in an iterative manner (Katagiri, *et al.*, 1998). It was shown that the solution obtained with the GPD algorithm converges to an optimal solution with a probability 1. Here we also use GPD algorithm. Denote $\nabla L(T;\Lambda)$ as the gradient of $L(T;\Lambda)$. Then the solution is obtained as,

$$\Lambda_{t+1} = \Lambda_t + k_t \nabla L(T;\Lambda)|_{\Lambda=\Lambda_t} \ , \tag{16}$$

with Λ_t being the parameters at the *t*-th iteration, and k_t being an update step size or learning rate. To speed up the convergence, a QuickProp algorithm is also applied to dynamically adjust the learning rate (Fahlman, 1998).

4. Multiclass Multi-Label Classifiers for TC

In this following, we apply the multiclass MFoM learning approach to TC. The system setup is similar to that for the binary classification (Gao, *et al.*, 2003).

4.1 LSI-based Feature Extraction

In IR, a multidimensional feature set is often extracted to represent a document. Each component of the feature set corresponds to the contribution of a term occurred in the document. In a typical application the lexicon usually has more than ten thousands entries. Many techniques, such

as feature selection (Sebastiani, 2002), have been proposed to reduce the dimension. Latent semantic indexing (LSI) (e.g. Bellegarda, 2000) is a way to achieve both feature selection and reduction. In this study, singular value decomposition (SVD) based LSI is used to decompose the well-known term-document matrix *H* into a multiplication of three matrices:

$$H = USV^T \quad , \tag{17}$$

$U : M \times R$ left singular matrix with rows $u_i, 1 \le i \le M$,

$S : R \times R$ diagonal matrix of singular values $s_1 \ge s_2 \ge .. \ge s_R > 0$,

$V : P \times R$ right singular matrix with rows $v_j, 1 \le j \le P$.

Both the left and right singular matrices are column-orthonormal. If we retain only the top *Q* singular values in matrix *S* and zero out the other (*R*-*Q*) components, the LSI feature dimension could be effectively reduced to *Q* which is often much smaller than *R*. By doing so, the three matrices are much smaller in size than those in Eq. (17) and it greatly reduces the computation requirements.

4.2 Baseline Multiclass Classifier

A document is represented by a multidimensional feature in the LSI space. The parameters of the classifier for each category are estimated from its training set. Here a linear classifier is used. The mean of the training vectors in each category is estimated as its initial parameters, i.e.

$$g_j(X;\Lambda_j) = \sum_{m=1}^{D} x_m . w_{jm} + w_{j0} \ , \tag{18}$$

where *D* is the feature dimension, $\Lambda_j = (w_{j0}, w_{j1}, \cdots, w_{jD})$ is the model parameter vector of the *j*-th category.

4.3 Learning Linear Classifiers with MFoM

In TC micro- or macro-averaging F_1, precision, and recall are often used to evaluate the system performance. Here we simulate the overall objective function with the micro-averaging F_1 defined in Eq.(11) for the training set. Similarly, other performance metrics can also be used. So the objective is to minimize $L(T;\Lambda) = -F_1^u$ with the GPD algorithm summarized as follows:

- Initialize the parameters of the classifier.
- Iteratively update the classifier
  1. Calculate the gradient,

$$\nabla L(T;\Lambda)\big|_{\Lambda=\Lambda_t} = A \cdot (w_1 \cdot \Delta FN + w_2 \cdot \Delta FP) \tag{a}$$

where $A = 2/(2TP + FP + FN)$ (b)

$$w_1 = (TP + FP + FN)/(2TP + FP + FN) \tag{c}$$

$$w_2 = TP/(2TP + FP + FN) \tag{d}$$

$$\Delta FN = \sum_k \sum_{X \in T} \frac{\partial l_k(X;W)}{\partial w_{ji}(t)} \cdot 1(X \in C_k) \tag{e}$$

$$\Delta FP = -\sum_k \sum_{X \in T} \frac{\partial l_k(X;W)}{\partial w_{ji}(t)} \cdot 1(X \notin C_k) \tag{f}$$

$$\frac{\partial l_k(X;W)}{\partial w_{ji}(t)} = \begin{cases} l_k(.) \cdot (1 - l_k(\cdot)) \cdot (-x_i), & k = j \\ l_k(.) \cdot (1 - l_k(\cdot)) \cdot \dfrac{\exp(g_j(.) \cdot h)}{\sum_{m \ne k} \exp(g_m(.) \cdot h)} \cdot x_i, & k \ne j \end{cases} \tag{g}$$

  2. Update the parameters of the classifier

$$w_{ji}(t+1) = w_{ji}(t) + k_t \nabla L(T;\Lambda)|_{w_{ji}(t)} \tag{h}$$

- Stop the iterations when a predefined maximum number of iterations is reached.

A closer look at the above algorithm reveals that:

1. The training instances, which are closer to the decision surface (i.e. $d_j(X;\Lambda)$ is close to 0 or $l_j(X;\Lambda)$ close to 0.5), play much more important roles in learning (See Eqs.(e-g)). Generally the noisy or abnormal instance, which is a factor of over-fitting, is far away from their decision surfaces. So this property of MFoM could reduce their influence.
2. From Eqs.(a-d), it is clear that the adjustment quantity of the parameters fuses information from the training instances itself but also those from the instances of all other categories. It can partially explain the robustness of the multiclass MFoM. Even if a category has only a small number of training instances, it can still robustly learn its decision surface from the information provided by the false positive instances assigned to it. (See Eqs.(a-f)).
3. The sign of the summarized descent for the false positive is different from that for the false negative. It can also avoid over-fitting to some extent.

These properties explain that multiclass MFoM learning is more robust than the binary MFoM, especially for the categories with small training sets. In contrast, most of the existing learning algorithms are not capable of learning with a small number of training instances, although they work well for large training sets.

5. Experimental Results and Analysis

We evaluate our proposed multiclass MFoM learning algorithm on the TC task and report our experimental results with the Reuters-21578[1] (ModApte version). Many evaluation studies have been documented (Joachims, 2002; Lewis & Ringuette, 1994; Yang & Liu, 1999). Here we follow the experimental set in (Gao, *et al.*, 2003). After initial processing, we obtain a collection with 90 categories consisting of 7,770 training and 3,019 testing documents. The LSI feature is extracted using SVD[2]. In this paper the LSI feature vectors, with the full rank of 1,613, are used. In the following experiments of MFoM learning, a is fixed at 60, b is 0.1, and h is 7.0. The top-15 category candidates are verified. Top-20 category candidates, except for the top-15 candidates, are used to

[1] http://www.research.att.com/~lewis/reuters21578.html
[2] http://www.netlib.org/svdpack/

describe the competitive model. The maximum iteration number in GPD is 100. All parameters are empirically set.

5.1 Properties of MC MFoM Learning

In this section we will study some properties of the MC MFoM learning. Figure 1 illustrates the convergence property of the GPD algorithm. Note that the empirical micro-averaging F_1 in the training set is shown, instead of the value of the overall objective function. The only difference is the sign of the value. It illustrates that the empirical micro-averaging F_1 in the training set is increasing in the iterative learning. After 50 iterations, a stable stage is reached. At the same time, the real micro-averaging F_1 for the training set reaches 0.978 from its beginning value of 0.520, obtained with the initial classifier parameters.

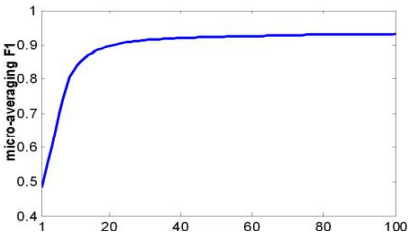


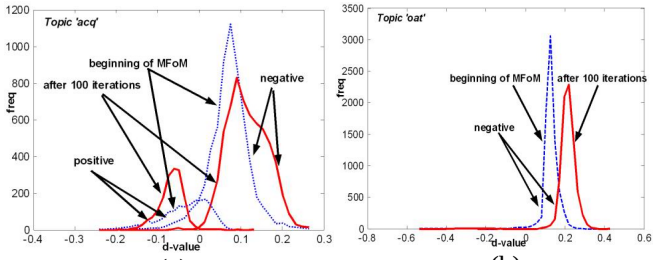
Figure 1 Convergence of GPD (x-axis: number of GPD iterations, y-axis: empirical micro-averaging F_1)

In Figure 2a we plot four training data distributions of the class misclassification function for the positive (i.e. the training instances labeled as category 'acq') and the negative (i.e. the training instances not belonging to 'acq') samples, respectively. Two distributions for the beginning of MFoM learning, and Two for the distributions after 100 iterations, are shown. Clearly MFoM learning reduces the overlap between the two competing curves, an indication that both false positive and false negative errors are reduced. At the same time, the curves become 'flat' after MFoM training, showing that the MFoM classifiers are more robust and less sensitive to data variation.

A similar set of curves for category "oat" is shown in Figure 2b. Here there are only 8 positive training samples, and therefore only the curves for the negative examples are depicted. It is interesting to note that even with so few positive training examples, we can clearly see the distribution curves for the negative samples move towards the right, indicating that a better separation in the training set is achieved after MFoM learning. We will also show later that the F_1 value for testing data also increased from 0.167 for binary MFoM to 0.500 for MC MFoM, corresponding to a significant improvement for this difficult case.

5.2 Effect of Small Training Sample Sizes

From the results shown in Figures 2(a-b), we can see that MC MFoM learning improves the distribution separation for the categories with both large and small training sets. The improvement should be more dramatic for categories with only few training examples to learn the corresponding classifiers. Table 1 lists a performance comparison between binary MFoM (Gao, *et al.*, 2003) and MC MFoM learning on 5 categories in which only less than 10 training instances are available. It is interesting to note that even with only one training instance, the MC MFoM classifier for the "sun-meal" category achieve a F_1 value of 0.667, much better than that obtained with the corresponding binary MFoM classifier. This microscopic performance analysis on TC is important because we are interested in designing robust classifiers in many new situations in which collecting a large training set could be expensive and difficult to accomplish.



(a) (b)

Figure 2 d-value distributions before and after MC MFoM learning for category 'acq' (a) and category 'oat' (b), only with negative samples shown (x-axis: d-value, y-axis: frequency)

Table 1 F_1 comparison for some rare categories

Category	# of Training instances	Binary MFoM	MC MFoM
Income	9	0.429	0.600
Oat	8	0.167	0.500
Platinum	5	0.286	0.833
Potato	3	0.333	0.750
Sun-meal	1	0.000	0.667

5.3 Comparison with Binary MFoM Learning

The binary classifiers are inherently more discriminative than MC classifiers because both positive and negative learning examples are used in designing the binary classifiers. Even with a simple baseline tree structure (Gao, *et al.*, 2003), the MFoM-trained binary classifiers performed very well in comparison with the top SVM classifiers (Joachims, 2002). With multiclass, multi-label classification, the MC MFoM learning described in Section 4.3 takes into account both the positive and negative training instances. We expect the MC MFoM-trained classifiers to perform at least as well as the binary MFoM-trained classifiers. A comparison of the top 10 categories among the linear SVM, binary and MC MFoM classifiers is listed in Table 2. Micro- and macro-averaging F_1 values for all 90 categories are also shown.

The results for binary MFoM in Table 2 are taken from (Gao, *et al.*, 2003), and those of linear SVM (J-SVM) are

taken from (Joachims, 2002, with C=1.0, which achieves the best micro-averaging F_1 on all 90 categories). Here the binary and MC MFoM classifiers used the full rank LSI feature and SVM employed all 9,600 features without any feature reduction. The full rank LSI-based linear SVM (LSI-SVM) is trained using the SVM-light[3] package with the default configuration and its results are shown together.

Table 2 The F_1 comparisons among linear SVM, binary MFoM, and MC MFoM classifiers for the top 10 and all categories

Category	J-SVM	LSI-SVM	Binary MFoM	MC MFoM
Earn	0.982	0.982	0.979	0.984
Acq	0.956	0.953	0.968	0.969
Money-fx	0.785	0.830	0.826	0.834
Grain	0.931	0.885	0.906	0.930
Crude	0.894	0.899	0.897	0.887
Trade	0.792	0.802	0.807	0.810
Interest	0.748	0.771	0.792	0.790
Ship	0.865	0.850	0.878	0.893
Wheat	0.868	0.797	0.870	0.844
Corn	0.878	0.800	0.891	0.889
Micro-avg (ALL 90)	0.875	0.866	0.884	0.888
Macro-avg (ALL 90)	NA	0.433	0.556	0.630

Table 3 Performance comparison on training sample sizes

Category split (threshold of training sizes)	Micro-averaging		Macro- averaging	
	Binary	MC	Binary	MC
All 90 (1)	0.884	0.888	0.556	0.630
Top 10 (150)	0.933	0.937	0.883	0.884
Other 80	0.720	0.738	0.512	0.598
Top 16 (100)	0.925	0.928	0.869	0.865
Other 74	0.679	0.712	0.483	0.579
Top 39 (30)	0.902	0.907	0.800	0.819
Other 51	0.537	0.588	0.361	0.486
Top 60 (10)	0.891	0.896	0.734	0.762
Other 30	0.310	0.451	0.198	0.357
Top 73 (4)	0.887	0.892	0.664	0.711
Other 17	0.138	0.377	0.094	0.269

Although the micro-averaging F_1 values are comparable for binary and MC MFoM learning, a closer look reveals that MC MFoM classifiers give a better macro-averaging F_1 value than that obtained with the binary MFoM classifiers. Next we study the performance difference on both types of categories with large and small numbers of training instances, respectively. These results are summarized in Table 3, starting with all 90 categories, splitting the categories into two collections according to the training sample sizes. For example, with a threshold of 4, we ended up with a split of 73 categories having more than or equal to 4 training samples and 17 categories with less than 4 training samples.

It is striking to observe the performance differences as a function of the training sample sizes. For small sizes, a significant improvement from binary to MC MFoM classifiers is clearly seen, even for micro-averaging F_1. For example, the value was improved from 0.138 to 0.377 for the categories with less than 4 training samples. Such a difference was not shown in Table 2 when comparing the top 10 categories. The above performance difference is even more pronounced when comparing the macro-averaging F_1 values. For instance, we see the values increase from 0.094 to 0.269 for the same collection of categories with small training sets. It suggests that a detailed sensitivity analysis as a function of the category performance is needed. We will discuss this next.

Table 4 Change rates for micro-averaging F_1

	Binary MFoM	MC MFoM
Micro-rate ($*10^{-4}$)	1.5382	1.5132

Table 5 Macro-averaging change rates (small TP)

Category	Binary MFoM			MC MFoM		
	TP #	FP #	Macro ($*10^{-4}$)	TP #	FP #	Macro ($*10^{-4}$)
Income	3	0	10.516	3	0	8.722
Oat	1	0	12.269	2	0	10.175
Platinum	2	0	10.516	5	0	8.722
Potato	1	0	24.537	3	2	24.376
Sun-meal	0	0	73.611	1	1	73.630

Table 6 Macro-averaging change rates (large TP)

Category	Binary MFoM			MC MFoM		
	TP #	FP #	Macro ($*10^{-4}$)	TP #	FP #	Macro ($*10^{-4}$)
Acq	689	15	0.104	687	12	0.086
Grain	130	8	0.511	132	2	0.418
Crude	179	31	0.428	177	33	0.361

5.4 Performance Sensitivity Analysis

We adopt a perturbation analysis by computing the performance change rate with respect to the category-specific *TP* (true positive) values for both micro- and macro-averaging F_1, they are computed as follows:

$$\frac{\partial F_1^m}{\partial TP_j} = 2\left[\left(\sum TP_j + \sum FN_j + \sum FP_j\right) \Big/ \left(2\sum TP_j + \sum FN_j + \sum FP_j\right)^2\right] \quad (19)$$

$$\frac{\partial F_1^M}{\partial TP_j} = \frac{2\left(\sum P_i\right)^2}{N\left(\sum R_i + \sum P_i\right)^2 (TP_j + FN_j)} + \frac{2\left(\sum R_i\right)^2 FP_j}{N\left(\sum R_i + \sum P_i\right)^2 (TP_j + FP_j)^2} \quad (20)$$

[3] http://svmlight.joachims.org/

where *N* is the total number of categories being averaged. It shows that the change rate in Eq.(19) is a constant across all categories for the micro-averaging F_1 as shown in Table 4.

On the other hand, the change rate in Eq.(20) for the macro-averaging F_1 is more complex. A few examples of the change rates are shown in Tables 5 and 6 for cases with small and large true positives, respectively. It is clear that the change rates of the macro-averaging F_1 for the small TP categories are larger than that of the micro-averaging F_1. While for large TP cases, the change rates are smaller for both binary and MC MFoM classifiers. The change rates with respect to other category-specific metrics can also be evaluated in similar ways.

6. Conclusion

In this paper, a multiclass, multi-label classification approach to TC is proposed. To take full advantage of both positive and negative training instances, a multiclass maximal figure-of-merit (MC MFoM) learning algorithm is introduced to train high performance MC classifiers. In contrast to the popular binary classification approach commonly adopted in the TC communities, the proposed MC scheme assigns a uniform score function to each category of interest for each given test sample, and thus the classical Bayes decision rules can now be applied. Since all the MC MFoM classifiers are trained simultaneously, we expect them to be more robust and work better than the corresponding binary MFoM classifiers, which are trained separately for each category and are known to give the best TC performance.

Experimental results on the Reuters-21578 TC task indicate that the MC MFoM classifiers achieve good improvements over the binary MFoM classifiers for both micro- and macro-averaging F_1 comparisons, especially for these categories with a limited number of training samples. For example in the cases with only less than 4 training instances, the MC MFoM classifiers give a macro-averaging F_1 of 0.269, which is significantly better than 0.094, obtained with the binary MFoM classifiers. Meanwhile, the increase in the micro-averaging F_1 goes from 0.138 to 0.377. Furthermore, for all 90 categories, mostly with large training sizes, the MC MFoM classifiers give a micro-averaging F_1 of 0.888, better than 0.884, obtained with the binary MFoM classifiers. It clearly shows that the proposed multiclass MFoM learning method achieves high performances and in the meantime is robust to training data variation. Extension from linear to other more sophisticate classifier structures should also give similar increased performance and enhanced robustness.

Acknowledgements

We would like to thank Dr. Qibin Sun for his support and the anonymous reviewers for their valuable comments.

References

Bellegarda, J. R. (2000). Exploiting latent semantic information in statistical language modeling, *Proc. of the IEEE*, Vol.88, No.8, pp.1279-1296.

Breiman, L., Friedman, J.H., Olshen, R.A., & Stone, C.J. (1984). *Classification and Regression Trees*. Wadsworth Int.

Dietterich, T.G. & Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263-286.

Fahlman, S. E. (1998). An empirical study of learning speed in back-propagation networks. *CMU-CS-88-162*.

Gao, S., Wu W., Lee, C.-H. & Chua, T.-S. (2003). A maximal figure-of-merit approach to text categorization. *SIGIR'03*, pp.174-181.

Joachims, T. (2002). *Learning to classify text using support vector machines*. Kluwer Academic Publishers.

Katagiri, S., Juang, B.-H. & Lee, C.-H. (1998). Pattern recognition using a family of design algorithm based upon the generalized probabilistic descent method. *Proc. of the IEEE*, Vol. 86, No. 11, pp. 2345-2373.

Kuo, H.-K.J. & Lee, C.-H. (2003). Discriminative training of natural language call routers. *IEEE Trans. Speech and Audio Processing*, Vol.11, No.1, pp. 24-35.

Lee, C.-H. & Huo, Q. (2000). On adaptive decision rules and decision parameter adaptation for automatic speech recognition. *Proc. of the IEEE*, Vol.88, No.8, pp.1241-1269.

Lewis, D. & Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. *The Third Annual Symposium on Document Analysis and Information Retrieval*, pp. 81-93.

Schapire, R. & Singer, Y. (2000). BoosTexter: A boosting-based system for text categorization. *Machine Learning*, Vol.39, No.2/3.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, Vol.34, No.1, pp.1-47.

Vapnik, V.N. (1998). *Statistical learning theory*, John Wiley & Sons, Inc., N.Y..

Weston, J. & Watkins, C. (1999). Support vector machines for multi-class pattern recognition. *ESANN*, pp. 219-224.

Yan, L., Dodier R., Mozer, M.C. & Wolniewicz, R. (2003). Optimizing classifier performance via an approximation to the Wilcoxon-Mann-Whitney statistic, *ICML'03*.

Yang, Y.M. & Liu, X. (1999). A re-examination of text categorization methods, *SIGIR'99*, pp. 42-49.

Zhang, J., Jin, R., Yang, Y.M. & Hauptmann (2003), A modified logistic regression: an approximation to SVM and its applications in large-scale Text Categorization, *ICML'03*.